Research Paper | Computer Science | E-ISSN No : 2454-9916 | Volume : 3 | Issue : 5 | May 2017

FREQUENT ITEMSET USING TRANSACTION REDUCTION TECHNIQUE IN DATA MINING

Ms. Sefali Patel [1] | Mr. Dheeraj Kumar Singh [2]

[1] Student,Information Technology, Parul University, Vadodara, India - 390006.

[2] Assistant Professor, Information Technology, Parul University, Vadodara, India - 390006.

ABSTRACT

Data reduction becomes a challenging issue in the data mining. Data reductions easily make the availability of the required space. Here analysis of the simple Apriori, partition based apriori and the apriori over reduction data set using the transaction reduction technique, existing reduction technique are not appropriate for data mining due to lack of consistency of maintaining the reduced data set. Three different approaches are proposed in which first is to sort the data set, the second one is grouping the data set and last is merging the data set. By performing this process it can define the time variation in processing time for large data set by using the reduction technique in this research an approach data Sorted merging reduction table Techniques are used by sorting and merging technique to solve this problem and reduce the data set. Then reduce dataset is maintained in the future as input of Apriori algorithm. The apriori algorithm is used to find the frequent item set in the data set. As simple apriori generates extremely large number of redundant rules which makes the algorithm inefficient and it does not fit in main memory. Therefore the proposed work will take care of these issues and will try to solve it.

KEYWORDS: Association Rule, Frequent Item Set, Support Count, Apriori, Transaction Reduction Technique.

I .INTRODUCTION

The more crucial and indispensable problem in numerous data mining applications are the frequent Item pattern mining, which is responsible for identifying the correlations, association rule analysis and many other substantial knowledge discovery tasks. Apriori is the most commonly used pattern mining algorithm that exploits breadth first search, bottom-up approach for each frequent item set. In general, the database transaction in Apriori is adopted by horizontal layout and the item set frequency is evaluated by counting its frequent appearance in each transaction. It gains good performance by reducing the size of candidate sets.

A. Association Rule Mining

Let I = {i1, i2, i3,..., id} be the set of all items in a market basket database and T = {t1,t2,t3,...,tn} be the set of all transactions. Each transaction ti contains a subset of items chosen from Item set I. A collection of one or more items is termed an item set. Support count is an important property of an item set. Support count refers to the number of transactions that contain a particular item set. Mathematically, the support count, σ(i), for an item set I can be given as follows:

$$\sigma(i) = |\{ti | X \subseteq ti,\ ti \in T\}|$$

An association rule is an implication expression of the form $A \rightarrow B$, where A and B are disjoint item sets, i.e., $A \cap B = \emptyset$ [12]. There are two important basic measures for association rules, minimum support and minimum confidence. Generally minimum support and minimum confidence are predefined by user/analyst so that the rules which are not so interesting or not useful can be ignored. Support of $A \rightarrow B$ is the total number of transactions where all items in A and B are together. Confidence of $A \rightarrow B$, determines how frequently items in B appear in transactions that contain A. The formal definitions of these two metrics are given below,

$$\text{Support}\ (A \rightarrow B) = \sigma\ (A \text{ and } B)$$

$$\text{Confidence}\ (A \rightarrow B) = \sigma\ (A \text{ and } B) / \sigma\ (A)$$

B. Frequent Item set Mining

Frequent pattern mining has been an important subject matter in data mining from many years. A remarkable progress in this field has been made and lots of efficient algorithms have been designed to search frequent patterns in a transactional database. Frequent pattern mining can be used in a variety of real world applications. It can be used in super markets for selling, product placement on selves, for promotion rules and in text searching.

II .RELATED WORK

A. Transaction Reduction technique for Improve Apriori algorithm

Mining of Association rules in large database is the challenging task. An Apriori algorithm is widely used to find out the frequent item sets from large database. But it has some limitations. It produces overfull candidates while finding the frequent item sets from transactions, i.e. the algorithm needs to scan database repetitively while finding frequent item sets. It will be inefficient in large database and also it requires more I/O load while accessing the database frequently. To solve the bottleneck of the Apriori algorithm, PAFI and Matrix based method used in proposed system.

1) Find Frequent Item Sets Using Apriori Algorithm

The most famous is the Apriori algorithm which has been brought in 1993 by Agrawal which uses association rule mining [6]. Association rules are usually required to satisfy a user-specified minimum support and a user-specified minimum confidence at the same time.

Association rule generation is usually split up into two separate steps:

1) Minimum support is applied to find all frequent item-sets in a database.

2) These frequent item-sets and the minimum confidence constraints are used to form rules.

Advantage of this algorithm, it is easy to find frequent item sets if database is small but it has two deadly bottlenecks. First, It needs great I/O load when frequently scans database and Second, It may produce overfull candidates of frequent itemsets.

2) Find frequent item sets using PAFI as well as Apriori algorithm [11]

D.Kerana Hanirex and Dr.M.A.Dorai Rangaswamy proposed efficient algorithm for mining frequent item sets using clustering techniques. They presents an efficient Partition Algorithm for Mining Frequent Item sets (PAFI) using clustering. This algorithm finds the frequent itemsets by partitioning the database transactions into clusters and after clustering it finds the frequent itemsets with the transactions in the clusters directly using improved Apriori algorithm which further reduces the number of scans in the database as well as easy to manage and available easily, hence improve the efficiency as well as new algorithm better than the Apriori in the space complexity but again it uses apriori algorithm hence efficiency not increase as much as required.

3) Improved the efficiency of Apriori algorithm based on matrix [10]:

Feng WANG and Yong-hua proposed an improved Apriori algorithm based on the matrix. To solve the bottleneck of the Apriori algorithm, they introduce an improved algorithm based on the matrix . It uses the matrix effectively indicate the affairs in the database and uses the "AND operation" to deal with the matrix to produce the largest frequent item sets and others. The algorithm based on matrix don't scan database frequently, which reduce the spending of I/O. So the new algorithm is better than the apriori in the time complexity but it is not suitable for large database.

4) Transaction Reduction In Actionable Pattern Mining For High Voluminous Datasets Based On Bitmap and Class Labels [9]:

Frequent pattern mining in databases plays an indispensable role in many data mining tasks namely, classification, clustering, and association rules analysis. When a large number of item sets are processed by the database, it needs to be scanned multiple times. Consecutively, multiple scanning of the database increases the number of rules generation, which then consume more system resources. Existing CCARM (Combined and Composite Association Rule Mining) algorithm used minimum support in order to generate combined actionable association rules, which in turn suffer from the large number of generating rules.

Explosion of a large number of rules is the major problem in frequent pattern mining that adds difficult to find the interesting frequent patterns.

III. PROBLEM DEFINITION

Data reduction provides a good solution which can lower the required space. Apriori employs an iterative approach known as a level wise search, where k-item sets are used to explore (k+1)-item sets. First the set of frequent 1-itemsetsis found by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support. The resulting set is denoted L1. Next, L1 is used to find L2, the set of frequent 2-itemsets, which is used to find L3, and so on, until no more frequent k-tem sets can be found. The finding of each Lk requires one full scan of the database. As the number of database scans are more the time complexity increases as the database increases

A. Apriori Algorithm

Apriori algorithm consists of following two steps.

a) Self Join

b) Pruining

Apriori uses a level-wise search where K-item sets are used to find (K+1)-item set.

a) First, the set of frequent 1- item sets is found which is denoted by C1.

b) The next step is support calculation which means the occurrence of the item in the database. Here it is mandatory to scan the complete database.

c) Then the pruning step is carried out on C1 in which the items are compared with the minimum support parameter. The items which satisfy the minimum support criteria are only taken into consideration for the next process which are denoted by L1.

d) Then the candidate set generation step is carried out in which 2-itemset are generated, denoted by C2.

e) Again the database is scanned to calculate the support of the 2-itemset. According to the minimum support the generated candidate sets are tested and only the item set which satisfies the minimum support criteria are further used for 3-itemset candidate set generation.

f) This above step continues till there is no frequent or candidate set that can be generated.

B. Algorithm Apriori

1) L1= {Large 1-itemset};

2) for (k=2; Lk-1≠0; k++) do begin

3) Ck = apriori-gen(Lk-1); // New candidates

4) for all transactions t ∈D do begin

5) Ci = subset(Ck, t); // Candidates contained in 1

6) for all candidates c ∈ Ct do

7) c.count++;

8) end;

9) Lk = {c ∈ Ck | c.count ≥ minsup}

10) end

11) Answer = ∪kLk;

C. Partition Algorithm

Partition is a way to split tables, indexes, and index organized tables into smaller pieces called partitions. This partition algorithm approach is implemented and evaluated against Apriori algorithm. Partition algorithm concept has been proposed to increase the execution speed with minimum cost. Initially only for one time the database is scanned and separate partitions will be created for each sets of item sets, which is 1-frequent item set, 2- frequent item sets, 3-frequent item sets. Partition algorithm can reduce time and gains the performance, manageability. The candidate item sets generated at each step is reduced and the scanning time is also reduced.

D. Partition Algorithm for Frequent Item sets Algorithm

Begin Number of clusters (NOC) =count of transactions (COT)/N

// where N is random natural number

FOR i= 1 to NOC DO BEGIN

FOR each cluster Ci DO BEGIN

FOR each transaction t €D DO BEGIN

Find t such that t having highest number of items Put t in Ci

END

END

Return item set.

IV. PROPOSED METHOD

Most of the data occupy a large amount of storage space. it is beneficial to reduce the data size which makes the data mining process more efficient with the same results. The transactions of databases are one way to solve the problem. Proposed a new approach for processing the merged transaction database. It is very effective to reduce the size of a transaction database. Their algorithm is divided into data reprocess and data mining. Simple Apriori, and the Apriori over reduction data set using the transaction reduction technique, existing reduction technique are not appropriate for data mining Due to lack of consistency of maintaining the reduction data set.

A. Transactions Reduction Technique

The proposed algorithm focuses on reduction related transactions and building a quantification table for pruning candidate item sets that are impossible to become frequent item sets. Finally, an example is provided to show the processes of our method. Algorithms like transactions reduction are used to reduce the size of a database, they suffer the following problems:

1) In the data reduction phase, the original database cannot be recovered to support transaction updates.

2) In the data mining phase, a lot of candidate's item sets could be generated in a large transaction database.

Since both need to scan the database more than once, they have a much higher process cost. The first problem is due to the lack of rule or constraint in the process of merging transactions in the data reduction phase. Therefore, the reduction database cannot be enlarged to its original form. In addition, they don't use user-defined threshold to filter infrequent 1-itemsets from the original database. We present a novel approach which can follow:

- Support local transaction variation.
- Make the compressed database smaller than the original one.
- Reduce data mining time.
- Merge related transactions to generate the reduce database.
- Discover frequent item sets.
- Using our self use full data base.

B. Reduction Technique (DSMRTT)

We called our approach the Mining DSMRTT -Data Sorted merging reduction table

Techniques which has three phases:

1) Merge related transactions to generate a reduction database

2) Data sorting and merging.

3) Discover frequent item sets

Three different approaches are proposed in which first is to sort the data set, the second one is grouping the data set and last is merging the data set. By performing this process it can define the time variation in processing time for large data set by using the reduction technique in this research an approach called DSMRTT-Data Sorted merging reduction table Techniques are used by sorting and merging technique to solve this problem and reduce the data set. Then reduce dataset is maintained in the future as input of Apriori algorithm.

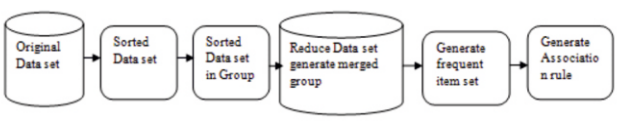

Fig 1: Block diagram of reduction Technique.

C. The Process of Database Compression

Let *d* be a relation distance and it is initialized to 1 at the beginning. Transactions will be merged into their relevant transaction groups in the merged blocks based on the transaction relation distance. M2TQT consists of the following steps:

Step 1: Read a transaction at a time from the original database.

Step 2: Record the information of the input transaction to build a quantification table.

Step 3: Compute the length *n* of the transaction.

Step 4: If the merged block is not empty, read the relevant transaction groups from the merged block.

Step 5: Compute relation distance between the transaction and relevant transaction groups. If the transaction is a superset of the longest transaction of a relevant transaction group, a subset of the smallest transaction of a relevant transaction group, or equal to one transaction of a relevant transaction group, it can be merged into the relevant transaction group.

For example, we assume *d*=1. Two transactions {BCG} and {BG} are merged into a relation transaction group {BCG=2.1.2}. A "=" symbol is used to separate items and their respective support counts. We read another transaction {BC} and compute the relation distance between {BCG=2.1.2} and {BC}. Since the relation distance is 1, {BC} is merged into the relation transaction group. Finally, the relevant transaction group becomes {BCG=3.2.2}.

Step 6: Compute the relation distance between the transaction and those transactions coming from *(n+d)* block, *n* block, and *(n-d)* block where $n > d$. If it finds the satisfied relevant transactions, merge the transactions to become a relevant transaction group and then classify it as *(n+d)* merged block, *n* merged block or *(n–d)* merged block. If no relevant transaction can be found, the transaction is classified as *n* merged block.

Step 7: Repeat the above six steps until the last transaction is read.

Step 8: Read a transaction from the merged blocks.

Step 9: Compute the relation distance between the transaction and all other transactions in the relevant transaction groups. If the transaction is a sub-transaction of the maximum length transaction of a relation transaction group and its distance is satisfied, it can merge the transaction into the relation transaction group to generate a new count. The process continued until the last transaction is read.

Step10: Set *d* to *d+1*.

Step11: Repeat the above steps 8 - 10 until no more relation distance is found between transactions.

D. Pseudo Code of Merge Mining Algorithm

//Phase one

1. D^M = Compressed database
2. L_1^M – {large1 – itemsets in compressed database};
3. for(k=2; $L_{k-1}^{M+1} \neq \Phi$; k++) do begin
4. C_k^M = merging – gen (L_{k-1}^M);
5. for all transactions T_M^* D^{M+1} do begin
6. C_k^M = subset (C_k^{M+1} , t^*); // Candidate contained in t^*
7. for all candidates c* C_k^{M+1} do begin
8. c*.count = c*.count + min-frequency(c*); // the smallest item frequency in candidate
9. end
10. L_k^M = { c* C_k^{M+1} | c*.count ≥ minsup}
11. End
12. Answer = $U_k L_k^{M+1}$;

// Phase Two

1. D = original database
2. for all transactions T_N D do begin
3. L_k^M = subset (L_k^{M+1}, t) // Large itemset contained in t
4. for all large itemsets l^* L_k^{M+1} do begin
5. l^*.count ++;
6. end
7. L_k ={ l^* L_k^{M+1} | l^*.count ≥ minsup}
8. Answer = $U_k L_k$;

For example, *T*1= {A, B, C, E} and *T*2 = {A, B, C, D} are two transactions. *T*1 and *T*2 are merged into a new transaction *T*3= {A2, B2, C2, D1, E1}. shows an example of the sort grouping method and shows an example of data mining subprocess.

the process Data Sorted merging reduction table Techniques is used to find frequent item sets from the new transaction data. It is possible that frequent item sets generated in. For this reason, it needs to verify those frequent item sets by scanning data again. The process called merge-mining algorithm is used to find frequent item sets from the new transaction data Set. and in figure 2 generates new data set and apply to that a priori algorithm. When we give minimum support and minimum confidence the compressed transaction methods give faster and better result as compare to other frequent pattern miming methods.

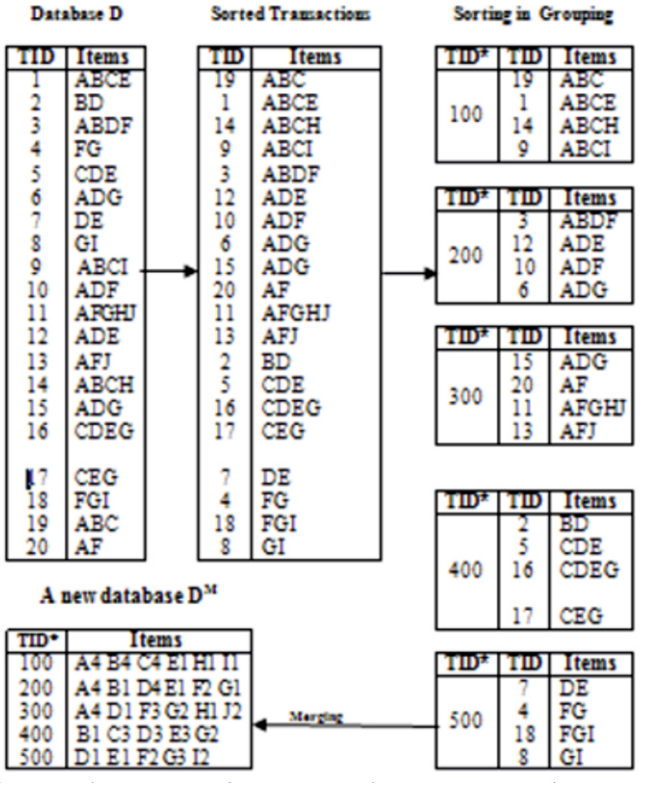

Fig. 2: Example of DSMRTT

V. RESULTS AND DISCUSION

A. Comparison on the basis of execution time and minimum support apriori, partition based, reduction based apriori

This table representing the number of records and execution time for Simple Apriori Algorithm, Partition based Apriori algorithm and Compressed Transaction based Apriori

Table 1: Execution Time of simple apriori, partition based apriori & compressed transactions based apriori with varying minimum

Minimum Support	Simple Apriori Algorithm (time taken in milsec)	Partition based Apriori algorithm (time taken in milsec)	Reduction Transaction based Apriori (time taken in milsec)
2	172	162	108
3	147	143	97
4	128	122	86
5	103	96	72

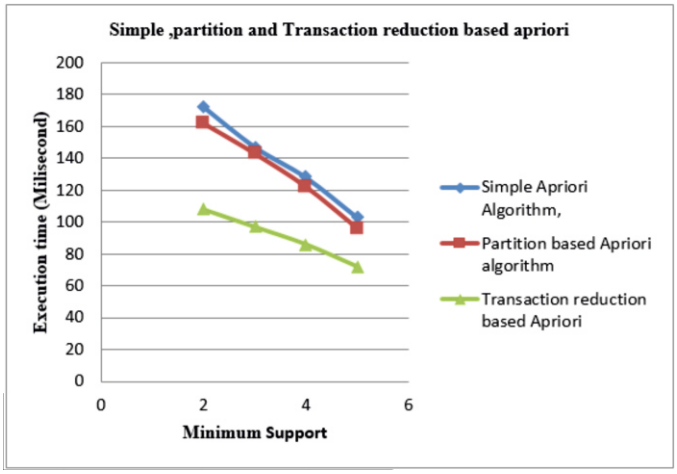

Fig 3 : Comparison Graph of Simple Apriori, Partition Based Apriori & Compressed Transaction based Apriori with varying minimum support

B. Comparison on the basis of execution time and minimum support apriorik, partition based, reduction based apriori

This table representing the number of records and execution time for Simple Apriori Algorithm, Partition based Apriori algorithm and Compressed Transaction based Apriori

Table 2: Execution Time of Simple apriori, partition based apriori & Compressed transactions based apriori with varying records

Number of records	Simple Apriori Algorithm (time taken in milsec)	Partition based Apriori algorithm (time taken in milsec)	Reduction Transaction based Apriori (time taken in milsec)
200	183	122	103
300	252	234	177
400	343	283	232
500	426	401	312

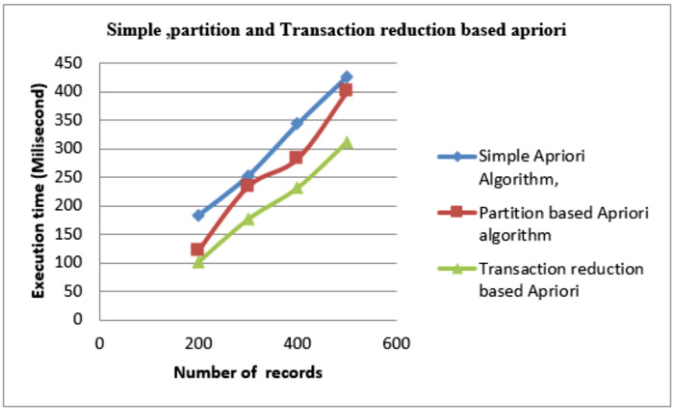

Fig 4: Comparison Graph of Simple Apriori, Partition Based Apriori & Compressed Transaction based Apriori with varying number of record

VI. CONCLUSION

The proposed work makes it clear that transaction reduction methods is much better as compare to the other methods a more efficient approach, called Mining Merged Transactions table technique is proposed, which can compress the original database into a smaller one and perform the data mining process efficiently. When we give minimum support and minimum confidence the compressed transaction methods give faster and better result as compare to other frequent pattern miming methods. Whenever the value of minimum support increases, the gap between our proposed and original Apriori algorithm decreases in view of time consumed. The time consumed to generate frequent item set in our proposed algorithm is less than the original apriori.

REFERENCES

[1] V.Vijayalakshmia, Dr.A.Pethalakshmib "An Efficient Count Based Transaction Reduction Approach For Mining Frequent Patterns" Published by Elsevier 2015.03.183

[2] Anjani Pandey, Gayatri Singh,"An Association of Efficient Mining by Compressed Database" Binary Journal of Data Mining & Networking 5 (2015) 33-35

[3] Mr.Vaibhav Kumar Sharma, Mr.Anil Gupta, Mr. B.L. Pal ,"An Efficient Approach for Data Mining with Compressed Data" International Journal of Computer Trends and Technology (IJCTT) – volume 28 Number 5 – October 2015

[4] K.Prasanna , Dr. M.Seetha , Dr. A. P. Siva Kumar "CApriori: Conviction Based Apriori Algorithm for Discovering Frequent Determinant Patterns from High Dimensional Datasets" International Conference on Science, Engineering and Management Research, IEEE 2014

[5] Dr.V.Vaithiyanathan, K.Rajeswari, Prof. Rashmi Phalnikar, Mrs.Swati Tonge "Improved Apriori algorithm based on Selection Criterion" Computational Intelligence and Computing ResearchIEEE 2012

[6] Loan T.T.Nguyen, Bay Vo, Tzung-Pei Hong,Hoang Chi Thanh,"CAR-Miner: An Efficient Algorithm For Mining Class-Association Rules,"Expert system With Applications 40(2013) 2305-2311, 2012@Elsevier Ltd. All Rights.

[8] Fan Zhang, Yan Zhang Jason Bakos," GP Apriori: GPU-Accelerated Frequent Itemset Mining". 2011 IEEE International Conference On Cluster Computing

[9] Anil Vasoya ,Dr. Nitin Koli "Mining of association rules on large database using distributed and parallel computing"Elsevier 2016.

[10] V.Vijayalakshmi,A.Pethalakshmi "A Performance based Transaction Reduction Algorithm for Discovering Frequent Patterns" International Journal of Computer Applications, 2014

[11] Rupesh Panwar, Prof. Abhishek Raghuvanshi "A Novel Transaction Reduction & Data Elimination Based Technique For Mining Frequent Item Sets From A Transaction Data Base" International Journal of Scientific Development and Research (IJSDR), 2016